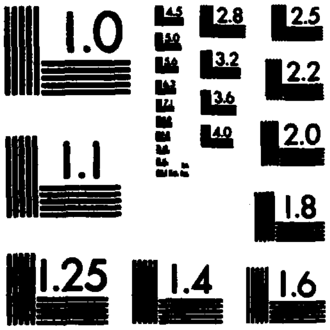
AD-A135 265 INTERNATIONAL CONFERENCE ON STIFF COMPUTATION HELD AT 1/1
PARK CITY UTAH ON A..(U) UTAH UNIV SALT LAKE CITY DEPT
OF CHEMICAL ENGINEERING R C AIKEN 31 MAY 83
UNCLASSIFIED AFOSR-TR-83-0971 AFOSR-82-0038 F/G 5/2 NL

END
FILMED
1-84
DTIC


1.0
4.5
5.0
5.6
6.3
7.1
8.0
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AD-A135265

(2)

AFOSR-TR- 83-0971

"INTERNATIONAL CONFERENCE ON STIFF COMPUTATION"

FINAL TECHNICAL REPORT

Contract No. AFOSR-82-0038

DTIC
ELECTE
S DEC 2 1983 D
H

DTIC FILE COPY

Approved for public release;
distribution unlimited.

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS
BEFORE COMPLETING FORM

1. REPORT NUMBER
AFOSR-TR- 83-0971

2. GOVT ACCESSION NO.
AD-A135 265

3. RECIPIENT'S CATALOG NUMBER

4. TITLE *(and Subtitle)*
INTERNATIONAL CONFERENCE ON STIFF COMPUTATION

5. TYPE OF REPORT & PERIOD COVERED
FINAL, 15 NOV 81-31 MAY 83

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)
Richard C. Aiken

8. CONTRACT OR GRANT NUMBER(s)
AFOSR-82-0038

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Department of Chemical Engineering
University of Utah
Salt Lake City UT 84112

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
PE61102F; 2304/A3

11. CONTROLLING OFFICE NAME AND ADDRESS
Mathematical & Information Sciences Directorate
Air Force Office of Scientific Research /NM
Bolling AFB DC 20332

12. REPORT DATE
31 MAY 83

13. NUMBER OF PAGES
55

14. MONITORING AGENCY NAME & ADDRESS *(if different from Controlling Office)*

15. SECURITY CLASS. *(of this report)*
UNCLASSIFIED

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

**Approved for public release;
distribution unlimited.**

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

The "International Conference on Stiff Computation" was held in Park City, Utah, April 12, 13, and 14, 1982, as announced in the brochure (Appendix I). This announcement was sent to approximately 1000 individuals; the names were acquired from a data base of authors of papers on the conference topic since 1972. An advertisement also appeared in BIT and SIAM newsletters.

The meeting was attended by approximately 45 individuals; about 33 of these presented a paper, of which approximately 20 were invited with (CONTINUED)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

83 11 29 202

ITEM #20, CONTINUED: various levels of support.

The purpose of the meeting was to examine state-of-the-art software development and theory for the numerical solution of stiff ordinary differential equations as it relates to application demands of today and tomorrow. A panel discussion at the meeting was particularly effective in advancing the objectives. Three very different viewpoints were expressed from the three separate (but overlapping) groups present: (1) Software developers generally feel the basic stiff problem (for which they have a very specific definition) is rather well worked out but additional difficult model features such as discontinuities, largeness, high frequency oscillation, etc., need special software attention. (2) Practitioners want more powerful and automatic software particularly for distributed systems modeled by PDE's, such as transport and reaction. (3) Theory developers are well behind what is needed from them but are advancing into nonlinear stability necessary for stiff method analysis.

Most of the better contributions to the meeting were reworked and will appear in text type-set form in a book titled, "Stiff Computation" due to appear from Oxford University Press in early 1984.

-a-

The "International Conference on Stiff Computation" was held in Park City, Utah April 12, 13 and 14, 1982 as announced in the brochure, labeled Appendix I. This announcement was sent to approximately 1000 individuals; the names were acquired from our data base of authors of papers on the conference topic since 1972. An advertisement also appeared in BIT and SIAM newsletters.

The meeting was attended by approximately 45 individuals. About 33 of these presented a paper, of which approximately 20 were invited with various levels of support. (See Appendix II for listing of speakers).

The purpose of the meeting was to examine state-of-the-art software development and theory for the numerical solution of stiff ordinary differential equations as it relates to application demands of today and tomorrow. We intentionally tried to have represented software developers, practitioners, and theoreticians approximately in equal numbers. This mixture proved to be effective in achieving our aim and for a very exciting meeting, as everyone with whom I spoke agreed.

A panel discussion at the meeting was particularly effective in advancing our objectives. A transcript to part of that discussion is attached in Appendix III. It was intersting to hear very different viewpoints from the three separate (but overlapping) groups present:

i). Software developers generally feel the basic stiff problem (for which they have a very specific definition) is rather well worked out but additional difficult model features such as discontinuities, largeness, high frequency oscillation, etc. need special software attention.

ii). Practitioners want more powerful and automatic software particularly for distributed systems modeled by PDE's, such as transport and reaction.

iii). Theory developers are well behind what is needed from them but are advancing into nonlinear stability necessary for stiff method analysis.

These observations are echoed in response to a questionaire that was sent out following the meeting (see Appendix III for answers to these). Particularly interesting was the lack of recognition of all attendees of the effect next generation computers would have on software development and advancement in

AIR FORCE OFFICE OF SCIENTIFIC RESEARCH (AFSC)
NOTICE OF TRANSMITTAL TO DTIC
This technical report has been reviewed and is
approved for public release IAW AFR 190-12.
Distribution is unlimited.
MATTHEW J. KERPER
Chief, Technical Information Division

our capability in solving stiff equations.

Most of the better contributions to the meeting were reworked and will appear in text type-set form in a book titled "Stiff Computation" due to appear from Oxford University Press in early 1984.

Accession For
NTIS GRA&I ☒
DTIC TAB ☐
Unannounced ■
Justification
By
Distribution/
Availability Codes

Dist	Avail and/or Special
A-1	

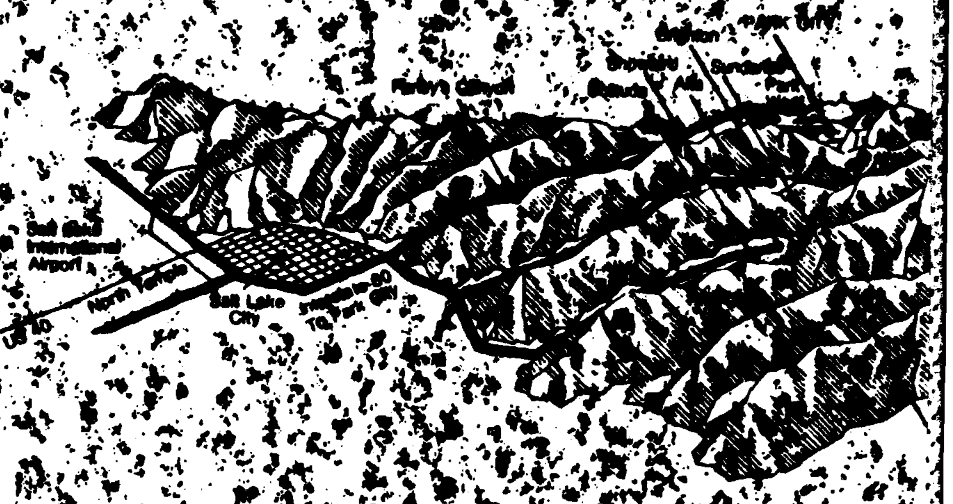
Ms. Mary Schmid
3095 MEB
University of Utah
Salt Lake City, Utah 84112

Reproduced from best available copy.

International Conference on Stiff Computation

April 12, 13, 14, 1982
at Park City, Utah

Sponsored by the U.S. Air Force
Office of Scientific Research


Salt Lake International Airport
North Temple
US 40
Salt Lake City
Interstate 80 To Park City
Parley's Canyon
Brighton
Park City
Snowbird
Alta

Location

This conference will be held in Prospector Lodge in Park City, Utah. Park City, once a boom mining town located high in the Wasatch Mountain Range and is now one of the world's great ski resorts. Attendees are invited to investigate "the greatest snow on earth" when spring skiing is at its best at Park City and nine other nearby resorts.

Park City is 25 miles east of Salt Lake City by six-lane Interstate 80, about 35 minutes from the Salt Lake City International Airport (See drawing above). Lewis Brothers Stages provides scheduled transportation from the airport to Park City.

Accomodation

Prospector Square Conference Center offers a 350-seat auditorium, a restaurant, lodging with over 275 sleeping rooms, and a complete health center. Attendees are encouraged to make reservations early to be guaranteed a room. Just fill out the attached registration form or contact Prospector Square Lodging, P.O. Box 1698, Park City, Utah 84060 Phone: (801)649-710

Purpose

The state of the art of software development and theory for the numerical solution of stiff ordinary differential equations will be examined as it relates to application demands of today and tomorrow. Priorities for future research and development directions will be indicated, based on users' computational needs.

Agenda

Three mornings and afternoons (April 12, 13 & 14) will be devoted to discussions and invited lectures by the international leaders in stiff computation theory and software development. In addition, representative users of stiff methods will present their perspectives. Invited speakers include:

- Professor G. Dahlquist (Stockholm, Sweden)
- Professor C. W. Gear (Urbana, IL, USA)
- Dr. A. C. Hindmarsh (Livermore, CA, USA)
- Professor, T. E. Hull (Toronto, ONT, Canada)
- Professor J. D. Lambert (Dundee, Scotland)
- Dr. W. Miranker (Yorktown Hgts, NY, USA)
- Dr. L. F. Shampine (Albuquerque, NM, USA)

APPLICATIONS AREAS REPRESENTED INCLUDE:

- Chemical Kinetics
- Combustion
- Atmospheric Phenomena
- Lasers
- Aerospace
- Electronics
- Control
- Highly Oscillatory Systems
- General Modeling Questions

METHOD-ORIENTED TOPICS INCLUDE:

- Choice of Best Method
- Packages & Package Development
- Basis of Method Comparison
- Automatic Stiffness Detection
- Large Sparse Systems; PDES
- Non-Linear Stability Analysis
- Parameter Estimation
- Singular Perturbations Methods
- Weighted-Residual Methods

Most lectures and discussions will be of a review nature rather than reports on latest research. However, several promising new approaches will be aired.

A syllabus containing a summary of each lecture will be available at the meeting. The proceedings will be printed by a leading publisher at a later date.

Call for Papers

Frequent users of stiff software are particularly encouraged to respond to this call for papers. If you would like to present a paper, please send an abstract to:

Professor Richard C. Aiken
Dept. of Chemical Engineering
University of Utah
Salt Lake City, Utah 84112

Selections will be finalized by January, 1982.

Registration

To Register for this meeting and to arrange a room at the conference location, please complete this form and send it to:

Ms. Mary [illegible]
3095 MEB
University of Utah
Salt Lake City, Utah 84112

CONFERENCE FEE: Attach a money order for $150 to be made out to "International Conference on Stiff Computation":

☐ Fee enclosed
☐ Bill me

ACCOMODATION: Please reserve a room with kitchenette at Prospector Lodge in Park City:

☐ A Single @ $54/night
☐ A Double @ $61/night

For the nights of: ____________

Name ____________

Address ____________

____

____

____

Telephone ____________

Reproduced from best available copy.

APPENDIX II

CONTENTS

VOLUME I

R.C. AIKEN, University of Utah:
STIFF REVIEW 1974-1982; I. APPLICATIONS
STIFF REVIEW 1974-1982; II. PACKAGES
STIFF REVIEW 1974-1982; III. THEORY (Listing)

VOLUME II

APRIL 12, 1982

(In Order of Presentation)

L.F. SHAMPINE, Sandia National Laboratories:

WHAT IS STIFFNESS?

F.T. KROGH, Jet Propulsion Laboratory:

NOTES ON PARTITIONING IN THE
SOLUTION OF STIFF EQUATIONS

G.D. BYRNE, Exxon Research and
Engineering Company:

ANOTHER VIEW OF STIFF
DIFFERENTIAL SYSTEMS

W.D. SEIDER, (speaker), C.W. WHITE, III,
G.J. PROKOPAKIS, University of Pennsylvania:

STIFF ORDINARY DIFFERENTIAL EQUATIONS IN
CHEMICAL PROCESS ANALYSIS

P.M. DEW (speaker),
T.S. CHUA, University of Leeds:

NUMERICAL INTEGRATION OF STIFF
DIFFERENTIAL/ALGEBRAIC EQUATIONS
WITH SEVERE DISCONTINUITIES

F.C. HOPPENSTEADT (speaker),
P. ALFELD, University of Utah:

EXPLOSION MODE ANALYSIS
OF AN H_2-O_2 REACTION

W.L. MIRANKER, IBM:

AN OVERVIEW OF THE HIGHLY
OSCILLATORY INITIAL VALUE PROBLEM

R.M.M. MATTHEIJ, Katholieke Universiteit,
Netherlands:

RICCATI TYPE TRANSFORMATIONS AND
DECOUPLING OF SINGULARLY PERTURBED ODE

R.E. O'MALLEY, JR., Rensselaer Polytechnic
Institute (speaker),
R.M.M. MATTHEIJ, Katholieke Universiteit:

DECOUPLING OF BOUNDARY VALUE PROBLEMS
FOR TWO-TIME SYSTEMS

T.A. BICKART, Syracuse University:

P-STABLE AND $P(\alpha,\beta)$-STABLE
INTEGRATION/INTERPOLATION METHODS
IN THE SOLUTION OF RETARDED
DIFFERENTIAL-DIFFERENCE EQUATIONS

D.J. Nolting, USAF Academy (speaker),
D.J. Rodabaugh, Lockheed:

STIFFLY STABLE LINEAR MULTISTEP METHODS

S.O. FATUNLA, Trinity College, Dublin:

P-STABLE HYBRID SCHEMES FOR INITIAL
VALUE PROBLEMS

APRIL 13, 1982

G. DAHLQUIST, The Royal Institute of Technology:

SOME COMMENTS ON STABILITY AND ERROR ANALYSIS FOR STIFF NONLINEAR PROBLEMS

W. LINIGER, IBM:

CONTRACTIVITY OF MULTISTEP AND ONE-LEG METHODS WITH VARIABLE STEPS

J.R. CASH, Imperial College of Science and Technology:

A SURVEY OF RUNGE-KUTTA METHODS FOR THE NUMERICAL INTEGRATION OF STIFF DIFFERENTIAL SYSTEMS

W.E. SCHIESSER, Lehigh University:

SOME CHARACTERISTICS OF ODE PROBLEMS GENERATED BY THE NUMERICAL METHOD OF LINES

B.A. FINLAYSON, University of Pennsylvania:

SOLUTION OF STIFF EQUATIONS RESULTING FROM PARTIAL DIFFERENTIAL EQUATIONS

S.W. CHURCHILL, University of Pennsylvania:

STIFFNESS IN HEAT TRANSFER

J.O.L. WENDT, University of Arizona (speaker)
W.A. HAHN, Exxon Production Research Center:

INTEGRATION OF THE STIFF, BOUNDARY VALUED ODE'S FOR THE LAMINAR, OPPOSED JET DIFFUSION FLAME

J.E. DOVE, University of Toronto (speaker),
S. RAYNOR, Harvard University:

A MASTER EQUATION STUDY OF THE RATE AND MECHANISM OF VIBRATIONAL RELAXATION AND DISSOCIATION OF MOLECULAR HYDROGEN BY HELIUM (abstract)

C.A. COSTA (speaker), M.Q. DIAS, J.C. LOPES,
A.E. RODRIGUES:

DYNAMICS OF FIXED BED ADSORBERS (abstract)

F.E. CELLIER, ETZ-Zurich:

STIFF COMPUTATION: WHERE TO GO?

VOLUME III

APRIL 14, 1982

C.W. GEAR, University of Illinois at Urbana-Champaign:

STIFF SOFTWARE: WHAT DO WE HAVE AND WHAT DO WE NEED?

W. H. ENRIGHT, University of Toronto:

PITFALLS IN THE COMPARISON OF NUMERICAL METHODS FOR STIFF ORDINARY DIFFERENTIAL EQUATIONS

A.C. HINDMARSH, Lawrence Livermore National Laboratory:

STIFF SYSTEM PROBLEMS AND SOLUTIONS AT LLNL

G.K. GUPTA, Monash University:

DESCRIPTION AND EVALUATION OF A STIFF ODE CODE DSTIFF

P. DEUFLHARD (speaker), G. BADER, U. NOWAK, Universitat Heidelberg:

AN ADVANCED SIMULATION PACKAGE FOR LARGE CHEMICAL REACTION SYSTEMS

L. EDSBERG, The Royal Insititute of Technology:

CHEMICAL KINETICS - AN OVERVIEW FROM THE POINT OF VIEW OF NUMERICAL ANALYSIS AND SOFTWARE IMPLEMENTATION

J. DEVOOGHT, Universite Libre de Bruxelles:

AN OVERVIEW OF STIFFNESS PROBLEMS IN NUCLEAR REACTOR KINETICS

S. THOMPSON (speaker), P.G. TUTTLE, Babcock and Wilcox:

THE SOLUTION OF SEVERAL REPRESENTATIVE STIFF PROBLEMS IN AN INDUSTRIAL ENVIRONMENT: THE EVOLUTION OF AN O.D.E. SOLVER

P.G. BAILEY, E.P.R.I. (speaker), P.V. GIRIJASHANKAR, D.L. HETRICK, W.N. KEEPIN and O.A. PALUSINSKI, University of Arizona:

MULTIRATE INTEGRATION ALGORITHMS APPLIED TO STIFF SYSTEMS IN NUCLEAR POWER PLANT SIMULATION

S.K. DEY, NASA-Ames Research Center:

APPLICATIONS OF PERTURBED FUNCTIONAL ITERATIONS TO NONLINEAR STIFF INITIAL VALUE CHEMICAL KINETIC PROBLEMS

J.-T. HWANG, National Tsing Hua University, Taiwan:

NONLINEAR SENSITIVITY ANALYSIS IN CHEMICAL KINETICS--THE SCALED GREEN'S FUNCTION METHOD

K.E. CHEN, Bethlehem Steel Corp. (speaker),
W.E. SCHIESSER, Lehigh University:

SOME EXPERIENCES IN THE SELECTION OF INTEGRATORS FOR LARGE-SCALE ODE PROBLEMS IN CHEMICAL ENGINEERING

8.1. The Questionnaire

Eight 'stiff questions' were asked of each attendee following the Park City International Conference on Stiff Computation. A total 36 written responses were received, representing 92 per cent of the attendence. The questions were intended to be important, general, and not of the type typically asked speakers during a conference presentation. The questions and quite candid responses follow:

QUESTION 1: Should the term stiff be reserved for the initial-value ODE problem with sharp initial transients or could it be used in a wider sense (differential algebraic, discontinuous, large sets, highly oscillatory, TPBVPs, PDE/BVPs with sharp transients, etc.)? Should another general term be defined? Should a number of terms, one for each specific type of difficulty, be defined? How important is terminology?

Bickart: *I tend to be against the coining of too many terms. I am quite content to have the term stiff mean, as I think it has come to mean, a dynamic system exhibiting both quickly and slowly changing modes in such a manner that a numerical solution process for its solution must be stable for all stepsizes to facilitate efficiency in the solution process.*

Byrne: *The indicated definition is really appropriate only for linear constant coefficient, ordinary initial-value problems (first-order equation or system of equations). Other definitions of stiffness are concerned with the size of the smallest time constant and the length of time of the phenomenon that causes the small time constant. For parabolic PDEs, stiffness can be* *attributed to a reactive source/sink term. I would not be uncomfortable with the use of stiffness for differential-algebraic systems (nonstiff integrator costs too much -- a pragmatic definition of a stiff system in any case). Of course, terminology is used in the ODE community to identify a specific problem type and software requirements. I feel definition is important.*

Cash: *I think that the term stiff should only be used for initial value ODE problems at present. Certainly, it should not be used so widely for two-point boundary-value problems as it is. When we define more precisely what we mean by stiff I see no reason why the term could not be extended to certain problems in the other classes you mention (particularly to parabolic PDEs). I feel that the correct terminology is extremely important, particularly to guard against using stiff solvers as black boxes on the wrong sort of problems.*

Cellier: *...What do you need the terminology for? I am rather pragmatic. I feel that the term should help people in solving their problems. Therefore, if I use the term freely to denote almost any integration problem which is numerically difficult to solve, the only thing a potential user can know is the fact that he has a difficult problem to solve and may be forced to consult a specialist. This may be useful information, but it may then be more straightforward to call this term 'difficult problem' rather than 'stiff problem'. If we restrict the term stiffness to mean only a particular class of difficult problems...this information is more useful, as the potential problem solver may then already know which integration technique to apply. Other classes of difficult problems (i.e., highly oscillatory problems) may also be defined, which again would tell the user which integration technique to apply. Ideally, an algorithm should be designed which can analyze a system description and find out for the user to which class of problems his system belongs...*

x133

Dove: *I suspect that the term stiff will be used in a broader sense whether we like it or not. From the user's viewpoint this poses real risks that inappropriate software will be selected for particular problems. For this reason, there may be real advantages in developing a more descriptive terminology for the various types of difficulty. The objectives of such terminology would include alerting the user to the fact that there are various kinds of difficulty and helping to guide him or her towards appropriate software.*

Enright: *There are two characteristics of stiffness that are essential. The first is that the mathematical problem is well conditioned and stable in the sense that small perturbations ~~in~~ in initial conditions or in the differential equation lead to small changes in the solution. The second characteristic is that when an attempt is made to integrate the problem with standard methods, a severe stepsize restriction results from the constraints of numerical stability. Without both of these characteristics a problem should not be considered stiff. For example, highly oscillatory initial-value problems are not stiff even though it is frequently the case thtat the inital conditions contain no high − frequency components and the latter characteristic is present. The former characteristic is not present as a small change in the initial conditions or in the accuracy requirements could result in a very different solution and, in this case, the stepsize of standard methods would be determined by accuracy rather than numerical stability.*

Finlayson: *I believe that a two-point boundary-value problem, when treated as a PDE that gives a set of equations whose eigenvalues are widely separated, (should be considered) a stiff problem. It might be useful to have a separate nomenclature, but such a problem qualifies as stiff according to criteria such as max* $|\lambda_i|/\min_i|\lambda_i|$. *Perhaps there should be sub-classifications under stiff.*

Gear: *It should be reserved for systems...for which the stable timestep of an explicit method is much smaller than that necessary to track the solution. This does not imply sharp initial transients, or vice versa.* $y' = -10^6(y-1)$, $y(0) = 1$ *is stiff but has no transient;* $y' = -10^{-6}e^{-10^6t}-e^{-t}$ *has a transient, but is not stiff.*

Hindmarsh: *'Stiff' should be restricted to initial-value problems for ODES, and possibly differential-algebraic systems, with one or more highly damped modes (and time constant small compared to the solution scale). Using 'stiff' for other types of difficulties only causes confusion.*

Krogh: *The only other place where use of the term might be justified is for boundary-value problems with sharp transients, or which would have sharp transients with a small perturbation in the boundary conditions. For the other case, the words you use are just fine.*

Mattheij and Soderlind: *Today it is widely recognized that stiff IVPs call for methods with special stability properties, i.e., the stability set should contain most of the left half plane. It is also well known that methods aimed at nonstiff problems (e.g., implicit Adams methods) for stability reasons are very inefficient when applied to stiff problems, despite the fact that they locally are highly accurate. Thus stiffness is a conflict between stability and accuracy requirements that appears in certain problems. Needless to say, definitions of stiffness that cannot account for this simple observation are not conceptually correct. Unfortunately, the classical definitions of stiffness fail because they are based exclusively on mathematical properties of the differential equation, and have thus caused severe misunderstandings. Obviously, a number of different terms, one for each specific type of difficulty, would be appropriate, as long as the difficulties have different origins. Note, however, that the mentioned conflict between* *stability and accuracy requirements is very general indeed. Since it may well appear in differential-algebraic, dicontinuous and highly oscillatory problems, there is no reason why the term stiffness should be not used in such cases. Even for certain classes of BVPs, a proper definition of stiffness carries over ad verbatim, although for general BVPS the notion of numerical stability seems far from being as well understood as in the IVP case. This is, however, probably due to the fact that numerical methods for BVPs in general are more complex than methods for IVPS. For this reason there is also a significant lag between the development of mathematical software for IVPs and BVPs.*

O'Malley: *The IVP people seem to have developed a special, nearly self-contained literature for ODEs which now includes a sequence of rapidly converging definitions of stiff (for their problems). The braoder computing public, however, views stiff problems as tough ones whose analytical solutions involve intervals of rapid transition and whose numerical solutions can easily require (due to stability considerations) use of very small stepsizes in regions where solutions are smooth. I prefer and would suggest using the term very loosely, recognizing that IVPs which are stiff 'in the technical sense' are best understood. I would not, in any way, restrict or deter research on grounds of established terminology or turf. In particular, I would like to emphasize the point that stiff BVPs for ODEs and PDEs are in very critical need of analytical and numerical study.*

Schiesser: *The term stiff should be reserved for the initial-value problem in ODEs with sharp initial transients. The clearest definition of a stiff problem for me is one in which the product of the problem time scale and the largest eigenvalue is large (much greater than one). All of the other problems mentioned in the question are important, but I think they should somehow be described with other words and terminology so we know what problems we're discussing. Otherwise, as mentioned at the conference, any difficult problem is termed stiff.*

Shampine: *I believe it is important to develop a reasonable terminology. It impedes research and causes users of software to select the wrong tools if the same term is used for fundamentally different phenomena. For example, I belive that the wide-spread belief that algebraic-differential systems are virtually the same as differential systems has caused both the difficulties I mentioned. Also, I believe that highly oscillatory problems are fundamentally different from stiff problems.*

Wendt: *The term stiff should NOT be reserved for initial-value ODE problems. A wide sense, certainly, involving boundary-value problems, must be included. Stiffness arises from certain aspects of physical problems. Flames...should be considered stiff, especially since attempts to solve (these) equations led to identification of stiffness as a mathematical problem.*

QUESTION 2: Has there been an interaction in the past between the hardware developers and the software (and theory) developers? Has this interaction or would such an interaction have been an advantage to either group? Will this interaction be important in the future; why?

Bickart: *I believe such interaction has taken place quite naturally, but largely informally. I see no special need to force it as concerns the numerical solution of equations for dynamic systems. Auxilliary array processors as a common-place item are on the way. I don't see a special need for auxilliary processors designed to implement a specific solution process.*

Byrne: *IEEE floating point chip design used by INTEL and others is an example (Kahan)...the Gleneder Beach, Oregon, conferences sponsored by DOE, LLNL, and Los Alamos National Laboratories bring users, manufacturers, and code writers together to discuss issues related to large-scale computing -- future and present. Of course it's important...*

Cash: *I believe that in the future there will be more interaction between hardware and software developers. About to appear is an important paper by Ower and Chershaw, who propose a new form of computer arithmetic to deal with error analysis. The interaction between software and theory developers is absolutely vital and of mutual benefit. On the one hand, we know that theory is unable to predict much of what happens in practice at present and software writers need to discover the way ahead by numerical experiment...*

Cellier: *Such an interaction has partly taken place -- and is indeed useful. To state an example: array processors have been developed which are useful tools but incredibly difficult to program without appropriate interface to the software side. In numerical integration of differential equations, many subtasks can be formulated as vector operations (using the same operation on different data, that is: a SIMD structure). It may thus be advantageous to compute these vector operations in parallel. This is, however, only possible if*

(1) ~~(a)~~ *the hardware specialist has designed his computer such that the parallel features can be addressed conveniently and efficiently, and*

2) ~~(b)~~ *the software designer makes use of these features by separating such vectorial operations in a modular way (e.g., by using LINPACK of which a version may be provided which makes use of the hardware vector operations).*

This is only one possible application. In principle, this is a question of properly designing the interfaces. Some interfaces may have to do with other pieces of already existing or still-to-be generated software, others with pieces of hardware.

Enright: *The evolution of vector machines is an exmaple of a hardware development that can be exploited by developers of* *packages for stiff problems but I don't feel that our needs are particularly special. Hardware capable of performing standard linear algebra operations is all that would be required.*

Gear: *Excepting Kahan (University of California at Berkeley), no.*

Gellinas: *Such interaction is extremely important for the future in view of the overriding importance that will become attached to linear solvers for sparse matrix systems which will be encountered increasingly in ODE/PDE applications. Such interactions will be even more crucial as multiprocessor computers become widely used. Critical opportunities for great advances in applications will be lost if software development continues to languish or die as has been the general trend recently.*

Hindmarsh: *On the level of computer arithmetic, there has lately been much such interaction, and it benefits both groups greatly. On higher levels, notably in numerical PDEs, there has also been much mutually beneficial interaction, with improved parallel/pipeline computers and software for them as results. Stiff ODEs that arise from PDEs are a subset of that area.*

Krogh: *Little interaction in the past...IEEE Floating Point standard is an exception and that because of an exceptional man, W. Kahan.*

Liniger: *I don't think there has been much and I doubt whether in the ODE area this will become very useful because of the general-purpose nature of our algorithms.*

Miranker: *Yes (to all three questions). The development of hardware depends on problem types (data types and operations).*

Pratt: *My knowledge is limited to having to structure code so that a vector processor can be employed; for example, Gauss-Seidel iteration is better (faster) than Jacobi iteration for solving systems of algebraic equations, but G-S is not vectorizable, while Jacobi iteration is, so that vectorized Jacboi is faster than G-S!*

Schiesser: *To the best of my knowledge, there has been no significant interaction between hardware developers and software developers with regard to stiff problems. Furthermore, I do not think there will be such interaction between the large mainframe manufacturers and people who work with stiff ODEs, simply because the stiff ODE problem is not important enough commercially to warrant special computer designs. Perhaps the microcomputer manufacturers will find the problem important enough to justify special designs, but I'm not sure small computers will ever be good for solving large, stiff ODE problems becuase they will be compute-bound; however, parallelism with small computers may be worth considering.*

Shampine: *I have not observed an interaction and I have no particular feelings about it. So far I have not noticed hardware designs calling for significantly different approaches. This could well change as parallelism is exploited, and then I would have to get involved. Even then I would expect to continue trying to make best use of the available hardware rather than influence its design.*

QUESTION 3: How automatic can and should a differential equation solver package be now and in the future? What advancements are seen in the near and far terms; how significant are they?

Bickart: *As automatic as possible. The solver built into the HP-34C is an example of what should be strived for in the more sophisticated software -- almost complete tolerance of user* *ignorance. Only when software suspects it is not handling a problem properly should it demand that the user become less ignorant.*

Brennan: *Designers should be aware of the many hidden dangers in totally automating a differential equation solver -- i.e., it may not be wise to implement software that requires little or no attention from the user. I currently maintain the integration package for a large trajectory simulation program (at the Aerospace Corporation). By a specialized input language, users simulate complex trajectories by specifying flight-dynamic and physical models with simple key words. Since the differential equations are automatically set up, the users may not even know how many differential equations are involved. The user is required to specify the type of integration method (Adams, RK, or BDF; fixed or variable step) although this requirement could conceivably be automated, too. There are dangers in automating too many features since the typical user fully accepts whatever the computer spills out as the correct answer...*

Byrne: *How portable is the first issue. I can visualize an ODE solver that would:*

(1) select machine parameters,
(2) select initial step,
(3) advise progress,
(4) switch methods (stiff, nonstiff),
(5) pick stepsize dynamically,
(6) report errors,
(7) graphics driven,
(8) graphics output, with zoom-in;

1-6 are here now, 7-8 are not quite here. How significant is it to look at gas kinetics with rates of 10^{10}? Pretty significant.

Cash: *I think that an ODE solver should be as automatic as possible. There are many black-box users who are only concerned with obtaining a solution to their problem with a certain number*

to 741

of correct digits, and it is important that we should satisfy such people. There are numerous advances to be made in ODE software -- Professor Gear's talk outlines many of these. Many of these advances will be very significant and in particular~~ly~~ I feel ~~that~~ control of global truncation error for stiff ODEs and automatic stiffness detection to be most pressing. Problems like the numerical solution of stochastic ODEs have not even got off the ground!

Cellier: *The more that can be made automatic the better. Keep in mind that more and more problems are no longer solved by specialists in numerical mathematics, but rather by engineers out in the field. These people need as much help as possible to get their problems solved quickly and correctly. Please keep in mind the gullibility of these average users. They believe in a result as soon as no error message is printed out. Automation helps to make sure that correct answers are produced if at all. I know that this opinion is about 180° different from the average attitude of a mathematician. The reason is that a numerical mathematician is proud of solving a problem as efficiently as possible. Automation included in the algorithm obviously slows down. However, for the average user, it is entirely immaterial if a solution of a problem requires 20 per cent more computing time if in return he gets more robustness. Automation helps also in tuning.*

Chen: *From a user's point of view, the more automatic and flexible, the better. I had occasion to use LSODE for the first time recently. It is so well documented and easy to use that I was successful on the first attempt...*

Churchill: *Automatic solvers are of very little interest to me. They seldom are useful for real problems. Special purpose methods are almost always more effective...*

#742

Deuflhard: *...Future advancements:*

(1) large systems, possibly in the linear algebra part of stiff solvers,
(2) stiff/nonstiff automatic forth- and back-switching,
(3) differential-algebraic equations with stiff or nonstiff ODE part.

Devooght: *My preference would be for a flexible approach where the user can decide what option to take. A code like LSODA which switches automatically between stiff and nonstiff methods is certainly useful but too much of a black box could be dangerous if one loses means to critically appraise the results.*

Edsberg: *According to my experience much of the software available in e.g., NAG, IMSL, and other libraries are hard to use for people from application areas. More robustness is needed and more needs to be done on the stepsize regulation. Perhaps much software is designed for a too narrow problem class (from the mathematical point of view).*

Enright: *The package should match the user's requested accuracy to the accuracy of the numerical solution in a method-independent way. Although we do not have this yet, it should be possible in the near future. A package should also recognize when it is inappropriate and report this to a user. For example, a nonstiff code should report to a user when a problem is stiff.*

Finlayson: *...need low-accuracy ones for 2-D, 3-D elliptic PDEs...*

Gear: *Depends on applications. Very automatic for simple uses, but very complex problems or those solved repeatedly will require hand tailoring for efficiency.*

Gellinas: *Automatic solvers can and should compile and solve automatically broad classes of ODEs and PDEs by simple user* *inputs. These solvers should provide for dynamic scaling of incommensurate variables and for a choice of alternative error norms. The significance cannot be overstated because current practices of exerting large, frequently redundant efforts on dedicated computer programs for each new application are unacceptably wasteful of both computer and human resources.*

Hindmarsh: *Levels of automation are continually advancing with research and experience. Near-term advancements include automatic method selection and Jacobian analysis (we have a version of the first item). Far-term automation will allow a user to get solutions on specifying nothing but the problem itself, i.e., nothing about solution method, tolerances, etc. This will impact the casual users greatly by reducing their setup time. But heavy users will still want to have their hands on their controls for optimal efficiency.*

Hwang: *A package of complete hands-off features sometimes may not be the best choice when computation speed is of prime concern. this is especially so when large-scale dynamic systems are being modelled. With a limited amount of user's effort, the solver may prove to be considerably more efficient. As long as user's interference is not annoying and tedious, a 'semi-automatic' package is perfect acceptable.*

Krogh: *I believe we have reached the point where the software should be completely automatic. But it should still allow the user to provide information about the characteristics of the problem if such information would be helpful. I believe there are still significant improvements in reliability to be made. Routines can be made more automatic and more flexible. Efficiency for general problems can probably be improved at least 30 per cent, with much larger improvements for some problems.*

Mattheij and Soderlind: *There is always a risk that general-purpose solvers tend to be no-purpose solvers. Today there exist* *very sophisticated general-purpose codes, some of which suffer from being too integrated in the sense that generality prevents them from being efficient for certain applications. An advancement that may overcome this difficulty would be to develop highly modular ODE solvers so that suitable applications software can be obtained by the change of a driver subroutine or a reconfiguration of subroutines in the package. Another very important improvement over existing stiff solvers would be to have more built-in reliability checks, e.g., to monitor the stability of the differential equation (which may sometimes be quite different from that of the difference equation). In particular, robustness is important for codes used as black boxes.*

O'Malley: *One can always find bizarre problems and counterexamples to ordinary experience. Thus, the need for special-purpose codes will remain. The aim is to obtain a differential equation solver which can seldom be tricked. One needs tough problems to challenge all-purpose packages and to suggest improvements of them. Substantial confusion now occurs because naive users like me don't understand what the TOL setting means. Much progress has already been made, but user reluctance to blindly accept output from current codes is healthy and justified. There will certainly remain plenty of advantages for special codes for many kinds of restricted problems.*

Petzold: *With few exceptions a differential equation solver should be as automatic as possible. The reason for this is mainly to relieve the users of the solver from having to worry about how the solver works as opposed to how best to model their problems. There are now solvers which diagnose stiffness and even switch to the most efficient methods for a given problem, but there are other difficulties which are not diagnosed very well by the current codes. For example, it would be useful to know whether a stiff code is inefficient (or fails) because of an inaccurate Jacobian matrix or because of a poorly conditioned*

Jacobian or too stringent error tolerances or frequent discontinuities in some derivative of the solution. At present, the diagnostics which are provided by stiff solvers are nowhere near as good as for the nonstiff codes.

Seider and White III: *...We envision a broadening in the scope of systems to be integrated, including systems with (1) discontinuities and (2) constraints on the variables. We also envision improvements that produce a smooth, differential result, as required by parameter identification algorithms that require derivatives without scatter...*

Shampine: *Wherever possible they should be made automatic. If additional information on the part of the user could affect decisions in an important way, the user should be able to influence the computation. I foresee codes which are rather successfull regardless of the type -- stiff or not. This will be a great convenience for users and may result in a net gain in efficiency. Perhaps more important is run-time monitoring of the computation to ascertain that the code being used was properly selected for the problem at hand and is being properly applied.*

Watts: *It is an important goal to strive for improving the robustness and capabilities of present-day ODE software. We should continue to provide packages which can automatically handle difficulties which arise frequently -- of course, research and current state-of-the-art techniques may not be sufficiently well advanced to completely relieve the user of some burdens. Software which automatically detects and copes with stiffness, type-insensitive software, will be the most important new development. We will also see capabilities, highly oscillatory problems, differential-algebraic systems, and low-accuracy solution requirements (such as with real-time or tabular data computations).*

Wendt: *It seems to me that initial-value problems are being beaten to death. In many combustion kinetics problems, the 1960s software of the NASA kinetics program is more effective than subsequent developments such as EPISODE. A moderate continuing effort to improve automatic solvers may be useful -- but there are many more important problems involving stiffness in BVPs that still require significant effort.*

QUESTION 4: Is theory for the mathematics of computation complementing well software development needs of today and tomorrow? What are the major needs and what are the prospects for filling these?

Bickart: *To the first question, my answer is yes. To the second question, my answer is colored by my relationship to electronic circuit analysis and design; we need an ever growing arsenal of tools for the analysis of large systems -- where large is always getting larger in people's minds -- of algebraic equations.*

Byrne: *Most theory supports linear ODEs or slowly varying nonlinear ODEs. We left them behind ten years ago. Need good work on error analysis, step selection, method switching. Prospects are slim, not many theoreticians have ever seen a real problem and universities won't support code work, in my opinion.*

Cash: *There has always been quite a wide gap between theory of computation and software development. It is important that they should come closer together but it is hard to see this in the near future.*

Cellier: *Principally, yes. I feel that a stronger interaction between numerical mathematicians and people from computer science (that is: software engineers) may be fruitful. However, even this interaction takes place more and more in that some people (like Alan Hindmarsh or Cleve Moler) are really both at the same time...*

Churchill: *Theory has never contributed significantly to numerical computation. Almost all advances have come from practitioners. I doubt that this will change.*

Deuflhard: *Theory is still open in extrapolation methods, implicit RK methhods, and stability. There is an unclear risk whether implicit RK methods or stability will help software developments -- at least unclear to me.*

Gellinas: *There is some complementation between theory and software in some areas where certain combinations of research talent have been assembled and when the right combinations of individuals decide to communicate closely and collaborate. There is a great need for supportive theory in PDE convergence and stability areas, particularly for nonlinear systems. Prospects for filling these needs in the short term are not too good: theorists frequently do not understand application needs and scientific practitioners frequently ignore theorists. Perhaps we can do beter over the long term.*

Hindmarsh: *The majority of places where theory is done still have a low regard for software, and vice versa. but the exceptions are growing in number and impact. Increased communication between both types, and also between them and those doing applications, is needed. The biggest problem is for theoreticians to learn of and address the features of realistic application problems.*

Krogh: *When developing software, I have usually found that the theory people did not provide what was needed. What theory I needed, I needed to do for myself. There are of course a very few exceptions. Major needs (some hope of being filled) are:*

(1) cheap and reliable global error estimation,

(2) good algorithms for boundary-value problems with very strong boundary layers,

(3) *effective tests for precision requests that are too stringent for the arithmetic being used.*

Liniger: *The problem is that the theory usually limps behind the software development and by the time the theory points out new algorithms and strategies, the (software) adopted earlier based on heuristics have become so widely used that it becomes very hard to convince people that something new and perhaps better is around.*

Mattheij and Soderlind: *There is still a significant gap between theory and practice. In terms of stiff solvers, many codes contain 'tuned' strategies and approaches that have not yet been justified theoretically. Thus the effects of variable steps on multistep integration procedures are not fully understood. Some of the major needs for theoretical improvement include: robustness, stepsize strategies, global error estimation and automatic detection of stiffness. Large problems and differential-algebraic systems also need further investigation.*

Miranker: *Yes. A major need is to make the computer itself a scientific instrument. One computes but the operations which a computer executes are not known to the user....Numerical analysis is essentially independent of computers. This is a concealed but treacherous gap and moreover, it can now be closed.*

O'Malley: *I feel that current software for boundary-value problems is woefully inadequate. Adaptive mesh generation is essential. Nonlinear problems require much more practical experimentation and theory. In the singular perturbations context, careful numerical experimentation can motivate the necessary theory and vice versa.*

Schiesser: *My impression is that there is really very little useful exchange between the people developing theory and those developing codes. We enjoy being with each other and drinking* *coffee at the breaks between papers, but the useful exchange of information seems limited. I could not help but notice the number of theorists who left the room when an applications paper was presented, and visa versa.*

Seider and White III: *We need better theory for the stability analysis of nonlinear systems, for improving estimates of the global truncation error, and for adjusting the stepsize and order of accuracy. The prospects of meeting these needs are excellent in the long term, but not promising in the immediate future.*

Shampine: *Many fundamental issues need attention. For example, variable-order codes are of the greatest importance but the theory describing them is, at best, fragmentary. I happen to be looking at the neglected areas of the effect of changing stepsize and of local error estimation -- a lot needs to be done.*

QUESTION 5: What are a few of the most important application areas that demand far more improved stiff solution techniques? How do you interpret 'most important' — by number of users, number of computer hours, the need for the solution (say for national defense)? Of these applications, how many are difficult because of the size or number of stiff sets to be solved?

Bickart: *Dynamic processes described by other than ordinary differential equations, such as voltera integral equations, functional differential equations, etc. At the present these tend to take too much computer time for their solution.*

Brennan: *At the Aerospace Corporation, we want to solve a differential-algebraic system of nilpotency three, but there is no software or known algorithm which can solve such systems. By solving this system, we could generate feasible state vectors during a trajectory from which a successful reentry of the space shuttle could be initiated. We have modified the problem to* *reduce the nilpotency to two, so we can obtain a solution using a modification of GEAR's algorithm. Initializing the variables to be consistent is also a problem. Software capable of solving general DAE systems (and initializing the variables) would be very useful in all prescribed path control problems.*

Byrne: *(1) Chemical kinetics, (2) enzyme kinetics, (3) circuit design. Chemical kinetics is part of reactor design, process design, air pollution studies, reactive fluid flow -- including combustion. I do not know how many users are concerned -- several thousand, I would guess.*

Cash: *I feel that the numbers of users and the need for a solution are the most important reasons. However, if there are uses for an application area, I also feel that the contribution made to the theory is also of great importance when new techniques are developed. There is also a need to take a special look at large systems arising from PDEs where things like storage of the Jacobian, which is perfectly OK for small systems, may not be possible.*

Cellier: *In fact, most difficult 'stiff' problems result from large systems. In particular, parabolic and/or hyperbolic PDEs translated into sets of ODEs by the method-of-lines approach are typical candidates of difficult problems. Parabolic PDEs result almost always in stiff sets of ODEs. Hyperbolic PDEs result in a different class in that some of the eigenvalues of the Jacobian tend to lie close to the imaginary axis and be complex. For this reason, the method-of-lines approach is known to be better applicable to parabolic PDEs than to hyperbolic PDEs. I suspect that this commonly stated sentence is not true. The method-of-lines could well be applied to hyperbolic problems if appropriate numerical integration techniques for this class of problems would be developed -- and I don't see any reason why this should be impossible. Up to now, such algorithms do not exist or have at* *least not been implemented in a general-purpose package...Typical areas: reactor kinetics, helicopter simulation (stiffness of the blades), and many more. I do not think that the number of users is really the important issue. Number of computer hours is often a key point. However, for some of the problems there exist alternative solutions (e.g., a piece of hardware simulating the behaviour in some sense). Some problems (like reactor kinetics) are key problems, because it is too dangerous to build these systems without knowing pretty well in advance what is going to happen.*

Chua: *These areas include the variable-step integration of differential-algebraic equations and the location of discontinuities. They are important because most large-scale practical problems can be modelled by a set of differential-algebraic equations containing frequent discontinuities.*

Churchill: *Split boundary-value problems, unstable behavior, multiple stationary states, recycle problems, partial differential equations, and integro-differential equations. Computational requirements and numbers of users. The inherent behavior, not the number of equations, is the primary source of difficulty.*

Deuflhard: *(1) Large chemical kinetics, (2) large circuit design, (3) inverse problems in chemistry and electronics, (4) many -- component systems; most important: contribution to progress outside mathematics, mainly in sciences.*

Devooght: *My only experience with stiff problems concerns nuclear reactor kinetics. In this field the situation is rather good because ad hoc methods have been devised taking into account the specific form of the differential system. My secondary experience concerns the integration of the Liouville equation for the density matrix in quantum mechanics. In this case the eigenvalues are widespread but close to the imaginary axis and*

BDF methods are in this respect very deficient. The integration can be very time consuming and the use of a good program would help atomic physics a great deal.

Dew: *There is still a long way to go before we get ODE integrations that can interface satisfactorily to PDE software based on the method of lines. The problem is selecting ODE integrations that can correctly match the stability of the PDE.*

Edsberg: *Chemical kinetics -- in which case the kinetics is only part of a larger model including, e.g., flow, diffusion, thermal exchange, etc. I think model building with chemical kinetics is of great importance for environmental research, e.g., atmosphere chemistry, combustion, pollution processes, etc.*

Enright: *Solving large systems of loosely coupled stiff equations is an area where improvements are required. This is an important area since many problems arise in a variety of areas and they are now expensive to solve. It is very likely that the special structure they possess can be exploited.*

Finlayson: *...time-dependent finite element codes -- criterion is computer hours.*

Gear: *Highly structured large problems arising from PDEs, large systems from networks, VLSI modelling. (The basis is) computer time.*

Gellinas: *Oscillatory systems. I interpret 'most important' in the context of breaking through important scientific problems which are presently unresolved because of inadequate numerical solution methods.*

Hindmarsh:

(1) *PDE-based problems, especially in 2-D and 3-D. Size is the main obstacle. Importance is due to number of problems, computer cost, and need.*

(2) *Systems with high-frequency oscillations where only an envelope is desired. Size usually not big. Importance due to numbers and cost.*

(3) *Systems with discontinuities (switches, etc.). Usually large size, but size is not the problem. Importance is due to numbers and cost.*

Hwang: *Computer simulation of complex chemical reaction processes (e.g., CH_4 oxidation), especialy at the initial stage where sensitivity analysis of the mechanistic models with respect to initial reactant concentration and uncertainties in rate coefficients is desired. Typically, these models involve* ~50 *components and hundreds of parameters. The rate equations are very stiff (stiffness ratio could easily reach 10^8 or higher). Solutions to such problems are important for a number of reasons: energy conservation, environmental protection, new energy resources, etc.*

Miranker: *Chemical reactions, large circuits, satellites, weaponry.*

O'Malley: *Large-scale problems are important. Further, the use of aggregation methods and other hierarchical techniques which allow one to build increasingly complicated models of physical problems are essential. Chemical kinetics provide a very important applications area. One needs to figure out at what level detailed kinetics are necessary, as well as when experimental detail is needed. The meaning of various steady-state models also needs clarification, as well as their use in computational procedures.*

Petzold: *I think one of the most important application areas involves the solution of relatively small systems of equations with modest accuracy repeatedly and inexpensively, for example, when the operator-splitting technique is used for solving PDEs, there often results a system of ODEs which must be solved at any* *mesh point and at every timestep. It is essential that this be done as quickly as possible, or else the computation becomes very expensive. This situation occurs in several diverse areas of application, including structural deformation problem and combustion modelling.*

Pratt: *In my field, no one is solving finite-rate chemical kinetics in gas turbine, piston engine, or power generation furnace simulation models because GEARB is not fast enough. I don't (know of) anyone but myself (who) is attacking that problem seriously at the moment.*

Schiesser: *The recent developments in ODE integrators have had a major impact on the computer-based solution of PDEs, and I think this will continue. In fact, I will go out on a limb and say that the numerical method of lines has the potential for replacing most of the classical methods for solving PDEs. The importance of PDEs seems evident, and I think we will see a growing use of PDE applications in industry as well as in academic research.*

Seider and White III: *Improved integration methods are definitely needed for:*

(1) Systems that pass through oscillatory regimes into the explosion mode, even for short periods of time; for example, in combustion and in limit cycles.

(2) Systems that involve combined integro-differential equations, as in heat transfer problems with radiation, conduction, and convection.

(3) Systems that are multidimensional; for example, 3-D natural convection.

(4) Systems involving determination of parameters to give a close fit to experimental data, especially when the integration results are very sensitive to small changes in the parameters or where discontinuities are encountered in the objective function with changes in parameters.

Shampine: *I am concerned about the convenience and reliability of solution -- all solutions would benefit from improvements in these areas. Comparison with the solution of nonstiff problems makes this clear although the latter would also benefit from research.*

Thompson: *Various simulation packages are starting to see use in the solution of really sophisticated problems, e.g., in the nuclear industry. As this trend continues, deficiencies in current methods (e.g., BDF) will become more apparent. As I see it, a crying need will surface for stiff solution techniques which better take into account the structure of large problems (e.g., sparsity, subsystems with different characteristics, partitioning, and global error control).*

Watts: *Chemical kinetics, elasticity-plasticity mechanical modelling, PDE modelling by ODEs, nuclear reactor analysis. Each of the criteria - number of users, expense, and need for solution -- defines a perfectly acceptable 'most important' label. The PDE modelling by ODEs leads to the largest class of equations to be solved.*

Wendt: *Boundary-value problems based on occurrences in nature, computer hours and storage required, need; combustion, energy, catalysts, etc. Size is a most important factor in the above.*

QUESTION 6: Can you cite examples where the stiffness is a necessary part of the mode; i.e., applications where the nature of the stiffness directly influences the objectives of the modeller?

Byrne: *Nonlinear models in chemical kinetics. If it isn't stiff, it isn't stable...*

Cash: *No!*

Cellier: *There exist some problems in the literature consisting of a fast and slow subsystem (especially in control literture). Although only the slow system response is important for the user, it is influenced by the fast modes as well. Some of the slow frequencies disappear if the fast subsystem is left out...*

Chen: *...A reactor with vastly different time scales linked in series is an example where stiffness is an inherent property of the entire system.*

Chua: No.

Deuflhard: *Parameter identification in chemical kinetics or electronics does require measurements in both the transient and the stationary phase of the process. Automatic simulation of mixed systems with fast and slow processes. Method of lines for parabolic PDEs.*

Dove: *There is large class of problems for which the ODEs of chemical kinetics must be integrated, often in conjunction with ODEs or PDEs representing, for example, transport processes in gases or gas-surface interactions. Examples abound in the chemical industry. In many cases, these systems of equations must be simultaneously solved at a very large number of points over a mesh in 3-D space. The main factor in the often very high cost of solving such problems is that of solving the large stiff systems of chemical kinetic equations. Thus there is great interest in simplifying such systems to their bare essentials, and in reducing the cost in other ways. Sensitivity analysis plays a very important role in such problems. There is, in my view, a potentially very large role in such areas as computer design combustion, e.g., of automobile engines.*

Edsberg: *Chemical kinetics. The use of e.g.~~,~~ the steady-state approximation in order to eliminate stiffness may be disastrous for following the correct solution trajectory, see e.g.~~,~~ TRITA-NA-8005, example 3, oxidation of propane, where iterations converge to wrong solution trajectory if the reduced problem is solved.*

Enright: *Stiffness is inherent in any model where transient behavio~~u~~r can affect the overall system behavio~~u~~r.*

Gear: *Some problems, e.g.~~,~~ relaxation oscillators, are limits of stiff systems as stiffness approaches infinity. Without knowledge of (the) stiff form, the behavior of the limit problem can't be determined.*

Gellinas: *Yes. Extreme, nonequilibrium behavio~~u~~r in most branches of the physical and chemical sciences (e.g.~~,~~ plasma physics, radiative systems, interactive fluids, combustion, and most 'strongly driven' systems) has frequently remained out of the reach of modellers really understanding essential physical/chemical processes because sufficienctly accurate, highly resolved numerical solutions of the defining PDEs/ODEs could not be attained.*

Hindmarsh: *Most chemical kinetics problems are necessarily stiff if the model is to be accurate. Steady-state assumptions are of unreliable accuracy and efficiency. We are at a point in the quality of stiff solvers where the modeller should NOT be influenced by the presence or absence of stiffness in forming the model -- only by its accuracy. I have been solving some oil shale particle models where NOT using ssa gave solutions over 400 times faster than the best one could do with the assumptions, and also solved the harder cases.*

Liniger: *Regions of rapid transitions in semiconductor equations, models in hyperbolic PDEs, chemical kinetics.*

O'Malley: *Power system modellers cite an example where traditional reduced-order models lead to mechanical failure, but where including highly oscillatory transients circumvent the failure and explain the earlier difficulty.*

Petzold: *This is certainly the case for combustion modelling, and the modelling of control systems.*

Pratt: *If stiffness is a necessary part of the model -- that is, if you wish to resolve the solution on the small time scales -- then the problem is only wasted effort on computing the slow modes...is the problem then properly called 'stiff'?*

Schiesser: *As the grid spacing in PDE solutions becomes smaller, the classical theory indicates that the resulting ODEs should be stiffer. However, we seem to have some evidenc that with adaptive regridding, and with the use of higher-order approximations for the spatial derivatives, this may not be the case. I think this is still an unanswered question that warrants some research. Also, the effect of nonlinearities on the apparent stiffness of ODEs is an important, unaswered question. We find computer run times to be extremely sensitive to small variations in nonlinearities.*

No italic

Seider and White III: *Most systems stiffen as they stabilize or approach a steady state. In many cases the model cannot be easily simplified as stiffness sets in, and stiffness is a necessary part of the model. In some cases, it is possible to simplify the model as the system stiffens, without loss of accuracy, to reduce* $|\lambda|_{\max}$ *and stiffness (see, for example, the fluidized-bed reactor model in our paper). Limit cycles, with regular oscillations over large time segments, are examples of systems that cannot be easily simplified when stiffness sets in. These systems stiffen periodically and model reduction would not apply at all times following the first onset of stiffness. In* *these cases, it seems impractical to periodically alter between models.*

Thompson: *Examples abound. Those with which I am most familiar involve simulation of nuclear steam systems, in which the treatment of kinetics is often of paramount importance.*

Wendt: *Detailed combustion kinetics. Remove the factors causing stiffness and you have thrown out the baby with the bathwater.*

QUESTION 7: Do you see an advantage to considering the interactions of model identification, parameter estimation, and solution? Be as specific as you can and please cite any literature in this area of which you are aware.

Byrne: *Suppose a reaction mechanism for a system is postulated for a general temperature range. Laboratory and literature data can then be used to develop a preliminary set of parameters for a generalized Arrhenius reaction form. The kinetics model can be solved (stiff ODEs) to obtain a set of computed data. The computed data is then matched against the observed data for one or more final (product) species. The parameters for the reactions can then be adjusted by an algorithm to get new rates. The system can be solved again, etc. until the computed and observed data match up. If the match-up is awful, then so might be the proposed model. I suppose the variance-covariance matrix could provide some idea of goodness. Usually not much helps unless the fit is right on.*

Cellier: *It is indeed realized meanwhile that a pure simulation tool is not that useful, as often no appropriate models are known (in particular in soft sciences -- so called 'ill-defined systems'). Only very recently some attempts have been made to automate the model building and model validation business. The research is here, however, still far from an answer. A good review may be:*

Vansteenkiste, G.C., and J. Spriet: 'Computer Assisted Modelling of Ill-Defined Systems' in *Progress in Modelling and Simulation*, F.E. Cellier, ed., Academic Press (1982).

In fact, the first half of this book deals with modern issues of computer-assistance in modelling. The second half deals with modern issues of computer simulation.

Churchill: *These interactions should be considered. However, parameter estimation without consideration of the uncertainty of the input is idle. See* Churchill, 'The Interpretation and Use of Rate Data,' Hemisphere, Washington, D.C. (1979).

Deuflhard: *Numerical parameter estimation techniques should also be able to monitor the model in terms of sensitivities of the model parameters. (see forthcoming proceedings volume:* P. Deuflhard, E. Hairer (ed): Numerical Treatment of Inverse Problems in Differential and Integral Equations (to appear in spring 1983).

Dove: *The use of sensitivity analysis, which deals with the interaction of the form of the model, the magnitudes of the parameters, and the effects of these factors on the solution, is absolutely crucial to problems in which chemical kinetics are important. There is a real need for professionally developed software packages in this area. We have had to write our own, so far! Leads to much of the literature in this area can be located under the names of H. Rabitz and K.E. Shuler. The topic is also important for the 'inverse problem' where one is given data on Y(t) and needs to find information about the underlying ODEs. For example, given experimental data on a chemical kinetics problem, what do those data tell us about the form of the model and about its parameters? There is already some software to deal with this, e.g., Curtis's CHEKMAT, but in my view a lot more work is needed. For example, very often the data available do not* *enable a unique determination of all of the parameters -- or even of any of them. Nevertheless, one would like to have a way of deciding, and expressing, the constraints which those data place on the possible values of the parameters.*

Edsberg: *Of course it is important to work interactively with a model, not only to solve one initial-value problem but to have also a qualitative idea of how the model behaves with respect to structure, parameters, initial values, etc. I (almost) agree with Aris in his book 'Introduction to the analysis of Chemical Reactors,' Prentice Hall 1965, pp. 325: 'It cannot be too strongly emphasized that it is folly of the first magnitude to approach the computer without first having as good a feel as possible for the structure of the problem.'*

Finlayson: *This is a very important area in engineering. Rarely do we know the parameters well enough to justify excellent mathematics solutions, but we'd like to be close.*

Hindmarsh: *I have no experience with model/parameter identification problems. But at present, some interaction seems wise. Solution accuracy requests should be as loose as possible when large model adjustments are being made. Methods and codes for calculating sensitivities with respect to parameters are useful here, and this is an active area.*

Liniger: *Yes, I think these questions are very important in simulation, e.g., in chemical kinetics, it is very hard to evaluate a model, even qualitatively, if one has little or no information about the rate constants. The solution may lie in a completely different ball park if those constants are off.*

O'Malley: *Certainly. Not enough talented mathematicians and engineers have faced the tough problem of parameter estimation. This intermediate link is critical to any such effort.*

~~51~~ 162

Petzold: *I think there is definitely an advantage to considering these interactions. For example, parameter estimation often involves the solution of several problems which differ from each other very little. This structure can sometimes be taken advantage of to speed the whole process. Parameter estimation procedures could also benefit from having solvers which produce solutions that have a smooth (or nearly so) behavior with respect to changes in initial conditions and other parameters.*

Schiesser: *Parameter estimation and model identification of PDE systems will become increasingly important in industrial applications and will serve as an important link between theory and experiments. Efficient methods for the repetitive solution of large sets of ODEs should therefore be pursued.*

Seider and White III: *Yes, George Byrne answered my question (Seider) by stating that discontinuities in the results of the numerical integrator cause problems for nonlinear programming algorithms that approximate a Hessian matrix; for example, using Powell's method. This problem needs to be resolved. However, in addition, methods to avoid a complete integration for each set of parameters should be examined.*

Shampine: *My own research has not been directed at such issues although I think about them from time to time. I do not see how one can avoid considering the interactions.*

Wendt: *Yes -- again with detailed kinetics. Sensitivity analysis and parameter estimation are most important.*

QUESTION 8: What will be the major advancements in simulation of stiff systems in the near and far terms? Are these of fundamental primary interest or are they secondary improvements?

Bickart: In the analysis of very, very large sets of equations -- really, the systems they describe -- mixed mode analysis and the concept of latency have come to the fore as a means of reducing the problem size to manageable proportions.

Byrne: Automatic stiffness/nonstiffness detection and method switching with dynamic memory allocation -- prototypes availabe now will become standard.

Cellier: ...Certainly, parallel processing may bring some key impulses into the game. Some problems may become solvable which are currently simply too expensive to solve. Whether the algorithms themselves may still improve very much, I do not know...Certainly, the software will -- and has to -- improve. In particular, the modularity of code has to be improved, and the interfaces have to be better defined (e.g., with respect to the data involved -- large systems mostly require a large amount of data to be entered; for this purpose, an appropriate interface to a data management system should be defined.

Churchill: I expect new special-purpose algorithms will be developed. They may in some cases involve general principles.

Edsberg: Easier-to-use software for nonexperts -- primary interest; automatic scaling of ODEs -- secondary interest; sensitivity analysis.

Enright: I see three distinct developments which are of fundamental importance:

(1) The development of more effective techniques which can exploit special structure such as only a few transients or weak coupling between subsystems;

(2) The acceptance by software developers of a uniform interpretation of accuracy. This will enable users to view packages more as black boxes;

(3) The acceptance and wide distribution and publication of

new improved packages as they become available. (This can best be accomplished through meetings such as this one).

Gear: *Better techniques for handling the linear algebra in very large problems. Techniques for decoupling subsystems.*

Gellinas: *Those associated with solving extremely stiff PDEs with newly emerging adaptive mesh techniques -- both near and far term. We are now seeing fundamental advances in this area which will require extremely robust ODE (and linear system) solvers.*

Hindmarsh:

(1) *Better automation.*
(2) *Larger computer capacity and higher speeds will help a lot even with present methods.*
(3) *Advances in sparse nonlinear algebraic system methods, with and without consideration of special machine architecture, will contribute greatly.*
(4) *Better interfaces with users, via discipline-dependent simulation software, will be of secondary importance.*

Krogh: *Better reliability and efficiency. Primary vs. secondary is hard to answer, but I'd probably choose the latter. But the improvements to be had are well worth doing.*

Mattheij and Soderlind: *...the gap between theory and software will decrease. Such advancements are important although they may not have a large influence on the software design. The importance is mainly from the robustness point of view -- a more rigorous theoretical framework is needed to sustain black-box use of the software, especially in nonlinear applications.*

Miranker: *...ill-conditioned problems generally (PDEs, integral equations, turning points, highly oscillatory)...*

O'Malley: *I believe extension of related research to partial differential equations and boundary-value problems will require fundamental new concepts.*

Petzold: *Some of the areas where we can expect (or hope) to see major advancements in the near future are:*

(1) solution of differential/algebraic systems, and more generally, constrained differential systems;
(2) solution of highly oscillatory ODEs;
(3) global error estimation for stiff ODEs;
(4) inexpensive solution of small systems which must be solved repeatedly;
(5) better diagnostic messages for stiff solvers.

(Of course, not all of these proboems are stiff, depending upon your definition.) These problems are of primary interest. It would require quite a few pages to list problems of secondary importance.

Pratt: *Special-purpose or hardwired computers, and rewriting or reselecting algorithms to take advantage of computer architecture; in short, doing for stiff systems what has been done for Fourier transforms in signal processing hardware!*

Schiesser: *...the development of integrators which can handle a changing number of ODEs during the solution to accommodate adaptive regridding in the solution of PDEs.*

Thompson: *The incorporation of stiff system methodology into widely available simulation packages, adequate implementation of root-finding techniques, valid automatic stiffness detection and method switching, and global error estimation, ~~is~~ of fundamental and primary importance. Potential advances in partitioning and subsystem identification are at least of secondary importance. Also of crucial importance are the needed improvements in documentation and the development of reliable, user-friendly software.*

55 964

Wendt: *I think that current technology on rapid Poisson solvers and techniques for handling large systems involving huge matrices will evolve from the classified literature and become very useful for stiff problems. I don't know much about them, but I hear -- through the grapevine -- that very powerful computational techniques have been developed by DOD and could be very useful in many other applications -- such as stiff boundary-value problems. These would involve radical changes in how stiff problems are attacked and solved.*

8.1.1. Comments on questionnaire (Kv)

Table 8.1 contains the number of responses to each question; generally, the most-answered questions were answered in greatest detail. The most popular -- and most emotional -- questions were Nos. 1 and 3. Opinions on question No. 1, concerning use of the word 'stiff' fell basically into two categories: practitioners who want a free hand with the useage, and developers/theorists who want a precise, restricted useage. The almost universal answer to Question No. 3 was that automatic packages should be automatic to the limit, with options for control; significant advances (in addition to ultimate robustness) are seen for special problem types.

TABLE 8.1

Number of responses to stiff questions

Question No.	No. of responses
1	35
2	28
3	36
4	27
5	30
6	21
7	18
8	27

Questions Nos. 2,4,5, and 8 drew about the same number of responses. From Question No. 2, we would conclude that there has not been much interaction between hardware people and software people, but the usefulness of such interaction is definitely recognized. It is clearly realized (Question No. 4) that theory lags well behind practice in stiff computation (unlike in many other areas of science). Question No. 5 resulted in a variety of most important application areas being mentioned, but particularly those involving partial differential equations and large numbers of equations; this was echoed in answer to Question No. 8 on what the major advances will be.

Questions No. 6 and 7 were not answered nearly as often as the others, and this could be significant. Generally, few respondees could cite cases where stiffness was necessary to the model (Question No. 6). The idea of 'system analysis' for stiff computation (#2.1) was definitely regarded as important, as was the parameter estimation problem, but these notions were considered new (Question No. 7).

A ninth question was asked concerning the usefulness of the Park City meeting. The consensus was that it was a very good idea to bring together individuals with widely differing backgrounds that share a common problem area. This would support the response sparsity to Questions Nos. 6 and 7.

8.2. Panel discussion at International Conference on Stiff Computation, Park City, Utah†

SHAMPINE: *We've heard during these talks so far quite a lot of computation described as stiff...anyone have opinions?*

GEAR: *Yes...the purpose of having a classification and giving names to classes is to group them into moderate size groups; if you say a problem belongs to class x, then you can say it can be solved by certain methods...it is madness to classify BVPs that require totally different techniques from IVPs as being stiff...BVPs are unstable as well as stable...in both directions, while stiff problems are strongly stable in one direction.*

MIRANKER: *This guy was walking down a street in New York and passed this store in which he saw a sign that said 'Stiff Differential Equation Solver.' He quickly ran into the store and said 'can you solve my stiff differential equation?' The storekeeper said, 'I don't know what you're talking about, I make signs!'*

SSS

SHAMPINE: *Many people think that our problems will be mopped up if we just build a bigger computer...or many more little tiny computers...what implications (from next generation computers) does this have on software and algorithmic developments?*

HINDMARSH: *There are going to be some minor changes as a result of new architecture, but I don't think they are going to be profound as far as the stiff computation problem is concerned. We are going to see higher speeds, certainly, and we are going to see far greater use of parallelism, and we will have to try to make use of that new architectural environment, but the obstacle*

---

†Moderator, Shampine; panelists: Byrne, Dahlquist, Gear, Hindmarsh

we refer to as stiffness will still be around...the most that will happen, I think, is that the boundary between mildly stiff problems and the stiff problem will move a bit. But the really stiff problems will be around.

ENRIGHT: *What about the emphasis of different codes? For example, extrapolation codes become more competitive as we get into parallel architecture...*

HINDMARSH: *They may very well...What I'm saying will not happen is we will fall back on nonstiff methods and do everything explicitly.*

PRATT: *...I've been trying for some years to solve large-scale systems...what I see happening is as we get bigger and bigger computers we can do things faster and faster, but we will always want to solve bigger problems and we need sophistication at the low end of the scale...*

BYRNE: *...there are two obvious directions that computing is moving...the day is not far off when...every engineering professional will have on his desk a mini which is linked to a supercomputer mainframe...this is mind-boggling...the mainframe is probably going to be a big parallel processor...we see in the literature snake algorithms...you put something on your mini and it snakes out for unused mini-computers in the network...you have all this computing power. What is the impact on stiff computation? There will be a change obviously in the algorithms. The linear algebra will be changed dramatically...we will see much more detailed modelling...*

SSS

SHAMPINE: *Many people feel (the stiff problem) is rather mopped up...I think we need to have a period of true confession...what areas do we not know what we're doing or don't know how to do*

40711

anything at all...this is an opportunity for you to say where you think things are going...(for example) Runge-Kutta methods for stiff problems, we're just beginning to explore them...doing an integration and locating where some function had a zero, our theoretical and practical understanding of that problem is almost zero. There are terrible difficulties in stating theoretically what it means to locate a root of a function when that function is being defined by the noisy process of solving a differential equation...it's easy to write down something that sort of works...there's one example of a true confession...

PETZOLD: *...I don't think anyone really has a good understanding of nonlinear differential algebraic equations, in general, and there's this whole question of turning points...*

CASH: *Some problems we can get nowhere with...a boat connected to the shore and the tension in the chain is some random variable, stochastic differential equations...*

SSS

SHAMPINE: *Stability analyses have been tilled since the earlier days...Some of us think this has very little to do with practice, but I have a feeling at the current time stability analysis is in a real state of ferment and that the theory is moving closer to the practice...*

DAHLQUIST: *I'm glad to hear you say that. Well, of course we try. All theory must be based on simplifications...see how much one can do with simple model problems...I think all progress must be based on this -- that one makes a theory or makes software for some standard situations...Henrici hoped when he wrote his book that the theory or the practice of differential equations could be handled just by the assumption that the stepsize times the Lipschitz constant should be chosen sufficiently small. Then it became more and more obvious that there were practical problems* *for which this was not the case. My own experience at the time was, at the beginning of the fifties, I encountered one problem where this difficulty existed. I tried to ask a lot of people from different application disciplines if they recognized this kind of situation and everybody said, 'No'. Around 1960, things became completely different and everyone became aware that the world was full of stiff problems. Even if a theoretician tries to get a good formulation of the real problems, it is sometimes really difficult to get them so well formulated that it can be a good starting point for a theoretical analysis...to return to differential algebraic equation analysis...it is still a little uncertain how many real nasty situations are left that require a general theory...a lot of practical situations particularly in classical mechanics are differential equations with inequality constraints...think of all the problems with Couloub's Law of friction...or two-body collisions, it is an inequality constraint that they are not allowed to penetrate...perhaps you should widen the discussion of differential algebraic systems also to cover the case of inequalities...I would really like to hear about other applications (of differential equations with inequality constraints)...*

SSS

SHAMPINE: *My impression is we have no satisfactory way of solving delay or difference differential equations...*

BICKART: *Some of the codes work well...as soon as you run into nonconstant delays you run into problems.*

SSS

KROGH: *I don't know what the theoretical problems are with handling discontinuities, but I haven't seen any practical difficulties. I have had features for handling discontinuities in my codes since 1969...*

SHAMPINE: *That's because you have infrequent discontinuities...*

KROGH: *The difficulty is in the restarting?*

SHAMPINE: *The order plummets. It has a disastrous effect if it happens often enough.*

SSS

SHAMPINE: *The inverse problem of parameter estimation is certainly a very difficult problem...*

BYRNE: *The difficulties are...the kinetics themselves are wicked...you start off with so-called literature rate coefficients which for the particular system at hand aren't really right so you're really off the mark for a starting point...generally you have interval constraints...we need a cheap nonlinear constrained least-square package...the problem is quite typically badly scaled...*

CHURCHILL: *It's an impossible problem. There is a non-uniqueness that will never be solved by any package...you never will have good enough information so that you can trust what you get...*

DOVE: *...virtually no kinetic experiment gives you information on one rate constant only...there is information on several constants and that information has to be unscrambled...*

PRATT: *...we mustn't forget the third leg of this thing and that is sensitivity analysis...without it you're groping...*

SEIDER: *We've had a great deal of success in tuning the parameters in (models of) non-ideal solutions with Powell's nonlinear programming algorithm written in 1977; it's very good* *for highly nonlinear functions as well as highly nonlinear constraints...*

BYRNE: *It doesn't work out for us...the line search is too expensive.*

SSS

SHAMPINE: *When we actually get down to solving a problem we have to choose a solver to do it...we have actually to pick one...how do we decide?*

ENRIGHT: *...we can certainly identify some codes that are not competitive...testing will allow us to choose between two algorithms that have a very similar design structure, but to compare two codes with very different structure I think is almost impossible unless we restrict the class of problems to a very small class that exhibits a special structure...*

PRATT: *...How do we chose between a BMW and a Vespa?*

BYRNE: *I think we all have built-in prejudices...if you use a code and you're not comfortable with it, you're not going to use it again.*

ENRIGHT: *...the problem here is that codes aren't trying to do the same things. Twenty years ago they were. Fixed stepsize we could compare...now codes determine accuracy requirements in a completely method-dependent way...*

ST 762

END

FILMED

1-84

DTIC